07

Hello World

Transcripción

[00:00] Bien, conocemos la estructura del proyecto. Ahora llegó la hora de ejecutarlo. Para eso tenemos aquí el botón de Play que es Run Application. Lo que esto va hacer es ejecutar este método main que tenemos aquí en nuestra clase Java que nos vino autogenerada.

[00:19] Vamos a darle un vistazo aquí a esta clase. Primero que todo vemos la anotación Spring Boot Application. ¿Qué significa esto? Bueno, si tienen curiosidad como saben, apretando tecla “Command” y dándole clic, vamos a ir al código fuente de esta clase, de esta notación. Esta no es propiamente una interfase, es una notación, disculpen, no es una clase.

[00:40] Vemos que esto hereda de diferentes anotaciones, por ejemplo de SpringBootConfiguration, EnableAutoConfiguración, ComponentScan. Si ustedes ya son familiares con Spring, propiamente dicho Spring sin Spring Boot sino Spring como tal, Spring puro, ya deben saber más o menos en qué consiste el Componente scan, aquí tú le específicas que paquetes quieres que sean escaneados por Spring al momento de inicializar la aplicación.

[01:11] EnableAutoconfiguración si tú quieres ir más a profundidad le puedes dar también. Clic con Command y vas a ver que eso también jala de otra notación como AutoConfigurationPackage. Tú puedes ir mucho más adentro de la jerarquía de clases. Por ahora no lo vamos a hacer. No se preocupen. Es solo un vistazo.

[01:31] Vemos que esta interfaz nos da algunos mitos como el exclude, excludeName, scanBasePackage, en caso necesitemos especificar qué paquetes queremos que sean escaneados o ignorados también.

[01:46] Entonces por ejemplo, si tú quieres usar alguna de esas opciones, alguna de las opciones que están aquí, alguno de estos métodos, lo único que tienes que hacer es simplemente abres el paréntesis al costado de la aplicación, “Ctrl + espacio” y te van a salir ahí los mismos nombres de los métodos que has visto en el código fuente.

[02:06] Entonces por ejemplo, si le quieres poner un exclude, te va a abrir y automáticamente puedes llenar los datos aquí. No lo vamos a hacer ahora, no lo necesitamos, solamente estamos explorando lo que hay. La magia de Spring Boot ocurre aquí. Como les expliqué anteriormente antes se usaba lo que es el servidor de aplicaciones. Esto era colocar tu archivo.war en un servidor aplicaciones, ejecutarlo ahí y una vez ahí, ya tu aplicación era accesible desde cualquier web browser.

[02:38] Pero con Spring Boot esto es al revés, porque el servidor ya viene embebido, el servidor ya viene dentro de tu aplicación. Generalmente es Tomcat pero también puedes usar Jetty si quieres o algún otro servidor de aplicaciones ligero, no vamos a cambiar Tomcat desde que se usa por defecto aquí. Lo vamos a mantener. No necesitamos cambiarlo.

[03:01] Y el método main que como ya les expliqué, otros cursos de Java básico es el método básico para ejecutar cualquier aplicación desde línea de comandos. El primer método que se ejecuta va a ser el main.

[03:14] Desde aquí vemos que SpringApplication.run, run es un método estático, vamos a explorarlo. Entramos Y vemos que bueno, es un método sobrecargado, y tiene diferentes opciones como algunos argumentos que puede recibir, etcétera. Aquí por ejemplo en el run le estamos mandando el contexto de ApiApplication.class en los argumentos que le mandemos aquí.

[03:42] No se preocupen, es un típico método main normal, es una típica clase aquí con un método estático normal, no hay nada del otro mundo, pero si tienen curiosidad por saber cómo funciona esto por detrás de las cortinas como quien dice, son libres, de explorar el código.

[03:57] Sin más preámbulos vamos a darle play. Puede ser aquí o puede ser aquí también o puede ser aquí, el que más les guste. No se preocupen. Le damos play, vemos que está gustando y nos sale un error aquí Lombok requires enable annotation processing. Esto es porque Lombok lo que hace es generar los códigos usando annotations.

[04:18] Hasta ahora no hemos hecho nada, pero como estamos utilizando Lombok tenemos que habilitar esta opción. Entonces, vamos a decir que sí. Queremos habilitar Lombok annotation processing. Vemos que aquí ya nos sale que Java Annotation processing ya fue habilitado. Vamos a ejecutar nuevamente proyecto para ver si este error ya no sale.

[04:40] Ahora nos da otro error más que dice Upgrade Module SDK in project settings to 17 or higher. Esto me quiere decir que el módulo que yo estoy usando actualmente no está con Java 17, está con Java 11. Entonces lo que yo voy a hacer es actualizarlo a Java 17. Bien, ya tengo el JDK de Java 17, entonces aquí le voy a dar okay.

[05:15] Y vemos que nuevamente comienza el proceso de reindex de mi proyecto. Reindexación completa nuevamente, vamos a ejecutar el proyecto que tengo aquí, entonces veo que Executing pre-compile, tareas, parsing Java y vemos aquí en nuestra terminal, déjenme agrandar esto un poquito y vemos aquí la terminal que comienza con algunos mensaje, por ejemplo aquí dice comenzando el ApiApplication usando Java 17 con PID alguna cosa. ¿Dónde está exactamente ahorita mi proyecto?

[05:53] Inicializado por mi usuario, está bien. No active profile set, falling back to 1 “default”. Eso significa que como no has especificado un Sprint profile,

automáticamente agarra el default. Eso está bien, no se preocupen, no es nada del otro mundo.

[06:09] Devtools property defaults active Set 'spring.devtools.add-properties' to 'false' to disable. Esto es por la dependencia que acabamos de agregar devtools. Y bueno vemos que Tomcat ha inicializado en el puerto 8080, bueno, va a inicializar Tomcat. Inicializando server, inicializando WebApplicationContext de Spring Boot.

[06:32] LiveReload está corriendo en este puerto, Tomcat inició satisfactoriamente en el puerto 8080 y la aplicación inició en 2.1 segundos. Son detalles de los logs, donde podemos ver todo lo que pasa dentro de Spring Boot, cómo es que genera el proyecto, cómo es que lo ejecuta y sin más que ver vamos ahora al browser, tenemos esto y vamos a entrar al localhost. Localhost:8080.

[07:06] Y nos topamos con un página de error. Pero esta página de error no es una página de error cualquiera, es una página de Spring. Si volvemos aquí vamos a ver que no hay nada en los logs y tampoco tenemos nada a nivel de la aplicación. No hemos quedado nada, por lo tanto es bueno que tengamos este error aquí. Significa que la aplicación está viva, está corriendo.

[07:33] Vamos entonces ahora a crear algún controller. Para eso creamos aquí una nueva clase, derecho new java class y le vamos a dar hello controller. Va a ser una clase solamente de ejemplo. Le damos aquí, perfecto y yo lo quiero en el paquete controllers. Me está marcando un error ahora. ¿Por qué? Porque obviamente el paquete controllers no existe.

[08:00] Entonces le voy a dar clic en el foquito rojo y le voy a decir: mueve esto al paquete 'med.voll.api.controller' el cual no existe pero lo va a crear automáticamente. Entonces vemos que allí está el paquete controller, HelloController, le damos clic aquí y comenzamos con el código que queremos hacer.

[08:30] Primero que todo, esto es un controller. Si no es familia con Spring Boot, no te preocupes, para decirle que es un controller, necesitamos primero anotarlo como controller. Controller y vemos automáticamente que jala de organización.springFramework.stereotype.

[08:50] Si se dan cuenta controller está en el paquete stereotype porque todos jalan de components. Esto es un poco de Spring básico, solamente para que lo tengan en cuenta y ya le especificamos que es un controller, pero esto era hasta las versiones antiguas de Spring porque ahora lo que tú tienes es RestController porque estamos quedando un controller de Rest.

[09:17] Y nuevamente si entramos aquí vemos RestController tiene también controller y ResponseBody, porque es un RestController. Como no retornamos HTML, ninguna página HTML, solo necesitamos retornar JSON, XML o texto o cualquier tipo de dato crudo. Entonces no necesitamos de un controller como tal, necesitamos también de ResponseBody para retornar ese tipo de datos.

[09:45] Le damos que queremos RestController, borramos esto y RequestMapping. ¿Eso para qué es? Para decirle más o menos qué ruta de HTTP está siguiendo este método que vamos a crear aquí. Por ejemplo yo le voy a decir que va ser “/hello”. Entonces, cada vez que yo entré a local host:8080/hello, debería hacer algo. ¿Cómo le digo qué hacer? Bueno. Tengo que crear el método.

[10:21] Eso va a ser un public String helloWorld() y retornamos un string que diga “Hello world!”; y finalmente su punto y coma. ¿Cómo esto va a funcionar? Porque aquí le voy a decir que esta ruta la mapea aquí con un método get. Si no están muy familiarizados aún con el tipo de métodos get, post, put, delete, no se preocupen. Vamos a aclararlo más a detalle en las siguientes clases.

[10:59] Por el momento lo único que tenemos que saber es que con GetMapping vamos a mapear el método en esta ruta para esta aplicación y lo que debería darnos es un hello world! Entonces ahora voy a guardar. Si se dan

cuenta guardé aquí. Voy a recargar aquí, sigue con el error. Ustedes me pueden preguntar: “Diego, ¿no se supone que hemos instalado DevTools y no deberíamos reiniciar la aplicación para que funcione? ¿Nos has engañado?” No, no es nada de eso, no se preocupen.

[11:35] Lo que pasa es que tenemos que configurar el proyecto para que pueda usar DevTools. Para eso vamos a las configuraciones del proyecto settings. En settings nos vamos a build, execution, deployment y aquí nos vamos a ir a compiler. Aquí vamos a marcar la opción Build project automatically, le damos check, y finalmente me estaba olvidando, tengo que ir a advanced settings y allow auto-make to start even if developed application is currently running.

[12:08] Esto va a habilitar que nuestra aplicación recargue automáticamente apenas guardamos, sin que tengamos que reiniciar el servidor. Entonces, le damos aquí, okay. Y debería ya estar listo. Para probar lo que voy a hacer aquí es detener mi servidor, ahora sí, manualmente lo detengo, vemos que el proceso ya tuvo un éxito. Ya terminó, ya está muerto, y voy a ejecutar nuevamente mi aplicación.

[12:35] Vamos ahí. Vemos que ya comienza a generar nuevamente los mismos logs que vimos anteriormente. Recargamos la página y tenemos que ir a /hello. Vemos aquí que dice Hello World! Entonces el código funciona ¿y el DevTools está funcionando? Vamos a ponerle “Hello world...” Vamos a guardar, vamos a recargar. No funciona. Okay, tenemos que darle un tiempo.

[13:17] Entonces, vamos otra vez y funcionó. Vamos a hacerlo otra vez solo para verificar que no pasó nada. Vamos a poner from Europe! Le damos a guardar. Le damos un tiempo. Yo no estoy haciendo absolutamente nada como pueden ver. Está aquí. Vemos que el servidor automáticamente reinició.

[13:38] Voy a recargar aquí y el código está actualizado. Entonces ya vimos cómo generar primero que todo. El ejemplo hello world!, las anotaciones de Spring, cuál es la jerarquía entre estas anotaciones, ya vemos de dónde sale

RestController, RequestMapping. Pueden estar confundidos ahora, pero no se preocupen.

[13:59] Esto fue el Hello world. En las siguientes clases vamos a explorar cada uno de los métodos de Spring, como el get, post, put, delete. Así como hay GetMapping, tienes PostMapping, PutMapping, etcétera, y lo vamos a ver mucho más a detalle. Como siempre les digo practiquen mucho, nos vemos en la siguiente clase.